CLAIMS

What is claimed is:

1. A system that facilitates processing of a document, comprising:

a host application that facilitates creation of the document; and

a programming component that at least one of embeds code in the document and links code to another document such that the document can be run independently of the host application.

2. The system of claim 1, the document runs on a client and a server.

3. The system of claim 1, the programming component facilitates scaling by providing more features when the document is running on a client and fewer features when the document is running on a server.

4. The system of claim 1, the programming component separates document information into data content and view content.

5. The system of claim 4, the view content maps programmable names to generic API (Application Program Interface) objects, which objects are exposed as view controls that can be programmed against.

6. The system of claim 4, the data content acts indirectly against the view content via data binding.

7. The system of claim 1, the programming component generates a data island in the document that is accessible by a client and a server.

8. The system of claim 7, the programming component automatically generates a data island in the document that conforms to a predetermined data schema and can be edited without the full host application running.

9. The system of claim 1, the programming component is event based such that the code runs according to an event that is related to a client or a server.

10. The system of claim 1 generates a runtime exception when a system error occurs.

11. The system of claim 1 controls permissions associated with the document according to whether the document is running on a client or a server.

12. The system of claim 1, the code includes data code portions of which are attributed to indicate if the corresponding data can be run on a client, a server, or both.

13. A computer according to the system of claim 1.

14. A computer readable medium having stored thereon computer executable instructions for carrying out the system of claim 1.

15. A system that facilitates processing of a document, comprising:
a host application that facilitates creation of the document; and
a data component that facilitates creation of a data island that is at least one of embedded in the document and linked to from another document.

16. The system of claim 15, the data island can be edited by running only a subset of components of the host application.

17. The system of claim 15, the data island can be accessed and modified on a server without having to start the host application.

18. The system of claim 15, the data island is synchronized with document contents when the document is run inside the host application.

19. The system of claim 15, further comprising a data model that is connected to the data island to work directly against data of the data island.

20. The system of claim 19, the data island is synchronized with document contents when the document is run inside the host application, and changes to the data model are moved into the document contents via a data binding mechanism.

21. The system of claim 15, data of the data island can be cached by marking the data using an attribute.

22. The system of claim 15, the document is one of an OLE structured document, an XML file, and a binary file that facilitates storing a persisted state of cached data, wherein if the document is a binary file, a reader/writer of the host application can be employed to insert the data island into the binary file and which reader/writer can be used to edit the data island.

23. The system of claim 22, when the OLE document is processed on either a client or a server, the cached data can be reconstituted out of the OLE document, manipulated, and changes to the cached data stored back into the OLE document.

24. A system that facilitates processing of a document, comprising:

a host application that facilitates creation of the document;

a programming component that separates contents of the document into a data model and a view model; and

a coupling component that couples the data model and the view model according to an environment in which the document is processed.

25. The system of claim 24, the coupling component data binds the data model to the view model, which data binding binds data elements of the data model to view elements of the view model.

26. The system of claim 24, the coupling component binds the data model to the view model when the environment is a client.

27. The system of claim 24, the coupling component binds the data model to the view model when the environment is on a server, only if a lightweight API for the view model exists that can be run on the server.

28. The system of claim 24, further comprising a host model such that when the document is processed by the host application, the host model, view model, and data model are invoked, and the coupling component data binds the data model to the view model and enables view control hookup of the view model to the host model so that the view model can be updated with data of the data model.

29. A computer-readable medium having computer-executable instructions for performing a method for processing a document, the method comprising:

providing a host application to facilitate creating the document;

providing a programming model that facilitates separating contents of the document into a view model and a data model;

creating a data island in the document; and

scaling features according to whether the document is being processed on a client or a server.

30. The method of claim 29, further comprising binding the data model to the view model, wherein changes made to the data island are moved into a host document that is running on the host application when the document is reopened by the host application.

31. The method of claim 29, further comprising configuring the programming model according to whether the programming model is executing on a server or a client.

32. The method of claim 29, further comprising invoking at least one of a host model, the view model, and the data model according to an event that is triggered.

33. The method of claim 29, further comprising processing data of the data island on a server without invoking the host application.

34. The method of claim 29, further comprising limiting permissions of the document according to whether the document is run on a server or a client.

35. The method of claim 29, the document is associated with at least one of a word processing application and a spreadsheet application.

36. The method of claim 29, further comprising embedding code in the document such that portions of the code can be attributed to indicate if the code is allowed to run on a client, a server, or both the client and the server.

37. A system that facilitates customization of a document, comprising:
means for preparing the document;
means for separating view content and data content of the document;
means for embedding code in the document;
means for creating a data island in the document; and
means for scaling features according to an environment in which the document is processed.

38. The system of claim 37, further comprising means for processing at least one of events and runtime exceptions according to the environment.

39. The system of claim 37, further comprising means for synchronizing contents of the data island with a host document when the document is run on a host application.

40. The system of claim 37, further comprising means for presenting the view content when the environment is on a server.